UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte Dennis Steven DeLorme; Margaret Rose Fenlon; Alan Leon Levering; Michael John Oberholtzer; Jeffrey John Parker; Richard Michael Theis; and Jerry Leroy Von Berge.

Appeal No. _____
Application No. 10/777,870

APPEAL BRIEF

Attorney Docket No. ROC920040005US1 PATENT
Confirmation No. 6113

CERTIFICATE OF ELECTRONIC TRANSMISSION
I hereby certify that this correspondence for Application No. 10/777,870 is being electronically transmitted to Technology Center 2169 via EFS-WEB, on June __, 2010.

/Charles R. Figer, Jr. / June 15, 2010
Charles R. Figer, Jr., Reg. No. 62,518 Date

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Dennis Steven DeLorme et al.	Art Unit:	2169
Application No.:	10/777,870	Examiner:	Paul Kim
Filed:	February 12, 2004		
For:	METHOD FOR SUPPORTING MULTIPLE FILESYSTEM IMPLEMENTATIONS		

Mail Stop Appeal Brief - Patents
Commissioner for Patent
P.O. Box 1450
Alexandria, VA 22213-1450

APPEAL BRIEF

I. REAL PARTY IN INTEREST

This application is assigned to International Business Machines Corporation, of Armonk, New York.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

III. STATUS OF CLAIMS

Claims 1-7 and 28-47 are pending in the Application, stand rejected, and are now on appeal. Claims 8-27 have been canceled.

IV. STATUS OF AMENDMENTS

There have been no amendments filed subsequent to the final rejection mailed January 15, 2010.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Applicant's invention is generally directed to a filesystem conversion process for converting from one filesystem type to a different filesystem type, which does not require shutting down the filesystem to perform the conversion. Para. [0010].[1] The invention ensures that all objects within the filesystem are converted and, from the perspective of a user, does not impact the performance and operation of the filesystem. *Id.* Filesystem types specify what information is stored for a filesystem object (e.g., a file, a directory), how that information is formatted, and how that information is arranged. Para. [0036]. This information is typically referred to as metadata. *Id.*

During normal operation of the computer system, the operating system and applications work together to invoke processes and routines that are aware of the underlying filesystem type and available metadata in order to properly create, delete, open and modify objects within the filesystem. *Id.* Changing from one filesystem type to another filesystem type generally involves changing the metadata (or its internal structure) that accompanies each object within the filesystem. *Id.* Conventional conversion methods require shutting down access to the filesystem during the conversion process. *Id.* The currently pending claims for converting from one filesystem type to another are not directed to the actual mechanics of translating metadata; but, instead, are directed to translating metadata while maintaining the filesystem in a full operational capacity.

For the convenience of the Board, claims 1, 32, 37, and 42, the independent claims have been reproduced below and annotated with references to the specification and drawings to satisfy the requirement to concisely explain the claimed subject matter:

[1] For convenience, all citations will be to the published application U.S. Publication No. 2005/0192918

Independent Claim 1

A method for maintaining a data structure corresponding to an object having a first link from a first directory and a second link from a second directory in a filesystem, Para. [0074] *the object to which the data structure corresponds being selected from the group consisting of a file and a directory in the filesystem,* Paras. [0062]-[0064] *the first and second directories being parent directories to the object to which the data structure corresponds* Paras. [0062]-[0064], [0074] and FIG. 10*, the method comprising the steps of:*

storing in the data structure a first anchor point for the object that references the first directory, said first directory implemented on a first filesystem type; Para. [0075]

storing in the data structure a second anchor point for the object that references the second directory, said second directory implemented on a second filesystem type different than the first; Para. [0075] *and*

concurrently with storing the first and second anchor points, converting the first filesystem type to the second filesystem type Para. [0036] *including activating the second directory and deleting the first directory* Paras. [0061]-[0069] and FIGS. 9A-9F *while maintaining the filesystem in a full operational capacity.* Para. [0036].

Independent Claim 32

A program product, Para. [0031] *comprising:*

a computer readable medium; Para. [0031] *and*

a program code Para. [0031] *configured for maintaining a data structure corresponding to an object having a first link from a first directory and a second link from a second directory in a filesystem,* Para. [0074] *the object to which the data structure corresponds being selected from the group consisting of a file and a directory in the filesystem,* Paras. [0062]-[0064] *the first and second directories being parent directories to the object to which the data structure corresponds,* Paras. [0062]-[0064], [0074] and FIG. 10 *the program code resident on the computer readable medium and further configured to store in the data structure a first anchor point for the object that references the first directory, said first directory implemented on a first filesystem type,* Para. [0075] *store in the data structure a second anchor point for the object that references the second directory, said second directory implemented on a second filesystem type different than the first,* Para. [0075] *and concurrently with storing the first and second anchor points, convert the first filesystem type to the second filesystem type* Para. [0036] *including activating the second directory and deleting the first directory* Paras. [0061]-[0069] and FIGS. 9A-9F *while maintaining the filesystem in a full operational capacity.* Para. [0036].

Independent Claim 37

A program product, Para. [0031] *comprising*

a computer readable medium; Para. [0031]

a data structure configured to be maintained by an operating system and corresponding to an object having a first link from a first directory and a second link from a second directory in a filesystem, Para. [0074] *the object to which the data structure corresponds being selected from the group consisting of a file and a directory in the filesystem,* Paras. [0062]-[0064] *the first and second directories being parent directories to the object to which the data structure corresponds,* Paras. [0062]-[0064] [0074] and FIG. 10*, the data structure comprising:*

a plurality of attributes related to the object; Paras. [0034], [0059], [0060], and [0079]

a first anchor point that references the first directory, said first directory being of a first filesystem type; Para. [0075] *and*

a second anchor point that references the second directory, said second directory being of a second filesystem type different than the first; Para. [0075]

program code resident on the computer readable medium and configured upon execution to access the data structure, and concurrently with accessing the data structure, further configured to convert the first filesystem type to the second filesystem type Para. [0036] *including activating the second directory and deleting the first directory* Paras. [0061]-[0069] and FIGS. 9A-9F *while maintaining the filesystem in a full operational capacity.* Para. [0036]

Independent Claim 42

An apparatus Paras. [0026]-[0029] *comprising:*

a processor; Para. [0027] *and*

a program code Para. [0031] *configured to be executed by the processor to maintain a data structure corresponding to an object having a first link from a first directory and a second link from a second directory in a filesystem,* Para. [0074] *the object to which the data structure corresponds being selected from the group consisting of a file and a directory in the filesystem,* Paras. [0062]-[0064] *the first and second directories being parent directories to the object to which the data structure corresponds,* Paras. [0062]-[0064], [0074] and FIG. 10 *the program code further configured to store in the data structure a first anchor point for the object that references the first directory, said first directory implemented on a first filesystem type,* Para. [0075] *store in the data structure a second anchor point for the object that references the second directory, said second directory implemented on a second filesystem type different than the first,* Para. [0075] *and concurrently with storing the first and second anchor points, convert the first filesystem type to the second filesystem type including activating the second directory and deleting the first directory while maintaining the filesystem in a full operational capacity.* Para. [0036].

Other support for the claimed subject matter may generally be found in FIG. 11 and the accompanying text at Paras. [0077] and [0078] of the published application, U.S. Publication No. 2005/0192918. In addition, it should be noted that, as none of the claims recite any means plus function or step plus function elements, no identification of such elements is required pursuant to 37 CFR §41.37(c)(1)(v). Furthermore, there is no requirement in 37 CFR §41.37(c)(1)(v) to provide support for the subject matter in the separately argued dependent claims, as none of these claims recite means plus function or step plus function elements, and so no discussion of any of these claims is provided.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 1-7 and 28-47are rejected under 35 U.S.C. § 103 (a) as being unpatentable over Cleraux et al. (U.S. Patent No. 6,944,620) (hereinafter "Cleraux") and in view of Jans et al. (USPG Pub. No. 2002/0188625) (hereinafter "Jans").

VII. ARGUMENT

Applicant respectfully submits that the Examiner's rejections of claims 1-7 and 28-47 are not supported on the record, are clear error, and should be reversed. All such claims have been rejected as being obvious over the prior art cited by the Examiner. Applicant respectfully submits that, in the instant case, the Examiner has failed to establish a *prima facie* case of obviousness as to the aforementioned claims, and thus, the rejections thereof should be reversed.

Based on the Supreme Court's decision in KSR International Co. v. Teleflex Inc., 127 S. Ct. 1727, 1734, 82 USPQ2d 1385, 1382 (2007), a *prima facie* showing of obviousness still requires that the Examiner establish that the differences between a claimed invention and the prior art "are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art." 35 U.S.C. §103(a). Such a showing requires that all claimed features be disclosed or suggested by the prior art. Four factors generally control an obviousness inquiry: 1) the scope and content of the prior art; 2) the differences between the prior art and the claims; 3) the level of ordinary skill in the pertinent art; and 4) secondary considerations of non-obviousness, such as commercial success of products covered by the patent claims, a long felt but unresolved need for the invention, and failed attempts by others to make the invention. KSR, 127 S. Ct. at 1734 (quoting Graham v. John

Deere Company, 383 U.S. 1, 17-18 (1966)) ("While the sequence of these questions might be reordered in any particular case, the [Graham] factors continue to define the inquiry that controls."). Moreover, in KSR, the Court explained that "[o]ften, it will be necessary for a court to look to interrelated teachings of multiple patents; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by a person having ordinary skill in the art, all in order to determine whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue" and "[t]o facilitate review, this analysis should be made explicit." KSR, 127 S. Ct. at 1740-41 citing In re Kahn, 441 F.3d 977, 988, 78 USPQ2d 1329, 1336 (Fed. Cir. 2006) ("[R]ejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness."). But, not every combination is obvious "because inventions in most, if not all, instances rely upon building blocks long since uncovered, and claimed discoveries almost of necessity will be combinations of what, in some sense, is already known." KSR, 127 S. Ct. at 1741.

As a result, after KSR, while there is no rigid requirement for an explicit "teaching, suggestion or motivation" to combine references, there still must be some evidence of "a reason that would have prompted a person of ordinary skill in the relevant field to combine the elements in the way the claimed new invention does" in an obviousness determination. KSR, 127 S. Ct. at 1731.

Applicant respectfully submits that, in the instant case, the Examiner has failed to establish a *prima facie* case of obviousness as to independent claim 1, 32, 37, and 42, and as such, the rejections thereof should be reversed. As a matter of law, the rejections of dependent claims 2-7, 28-31, 33-36, 38-41, and 43-47 necessarily must also be reversed. See, e.g., Hartness Int'l, Inc. v. Simplimatic Eng'g Co., 819 F.2d 1100, 1108, 2 USPQ2d 1826, 1831 (Fed. Cir. 1987) (dependent claims patentable if independent claims are patentable over the art). Although focus herein will primarily be on the independent claims, focus will be on the dependent claims where it is deemed necessary. Thus, Applicant's remarks in rebuttal to the Examiner's rejections are presented below, starting with relevant independent claims, and continuing with selected dependent claims that warrant separate mention. In some cases, specific discussions of particular claims are not made in the interests of streamlining this appeal. The omission of a discussion with respect to any particular claim, however, should not be interpreted as an acquiescence as to

the merits of the Examiner's rejection of the claim, particularly with respect to claims reciting features that are addressed in connection with the rejections applied to other claims pending in the appeal.

A. Claims 1-7 and 28-47 are not unpatentable over Cleraux in view of Jans

Independent Claim 1

As set forth above, Applicant's claim 1 is directed to converting a filesystem from one filesystem type to a second different filesystem type. The Examiner's primary reference, Cleraux, is directed to a method for supporting multiple filesystem types in a single mass storage device, where the system hosting the mass storage is able to access the files in each of the filesystem types. Cleraux uses the alternate, hosted, filesystems for devices that do not have mass storage devices of their own, but rather use the host system mass storage device as their filesystem. The other devices may use an alternate operating system, and therefore an alternate filesystem type than the host. Any features of the filesystems for the other devices that are unsupported on the host filesystem are written to and accessed through an emulation library allowing the hosting system to be able to access the file or feature being emulated.

Specifically, claim 1 recites "storing in the data structure a first anchor point for the object that references the first directory, said first directory implemented on a first filesystem type" and "storing in the data structure a second anchor point for the object that references the second directory, said second directory implemented on a second filesystem type different than the first." Cleraux fails to disclose an object having a first anchor point referencing a first directory on a first filesystem type and a second anchor point referencing a second directory on a second filesystem type different than the first. Rather, Cleraux discloses an object on a filesystem hosted on a mass storage device connected to a host system. *See* Cleraux, FIG. 2. The object is accessed by both the device without a mass storage device of its own, using the hosted filesystem, and accessed by the host system having the mass storage device. The host system accesses objects on the hosted filesystem through an emulation library if the hosted filesystem type is of a different type than that of the host system. Neither the filesystem for the host system, nor the filesystem for the device without a mass storage device, have an object with

two anchor points to two corresponding directories, each implemented on different filesystem types.

Claim 1 additionally recites "concurrently with storing the first and second anchor points, converting the first filesystem type to the second filesystem type . . . while maintaining the filesystem in a full operational capacity." Cleraux also fails to disclose or suggest converting a filesystem from one filesystem type to a second filesystem type while maintaining the filesystem in a full operational capacity. At best, Cleraux uses the emulation library in order to access files. Specifically, Cleraux discloses that the emulation library, in response to a request for a file, may be queried for information associated with that file. *See* Cleraux, col. 5, ll. 44-56. The emulation library, in turn, checks a repository for the file as well as a database for information associated with the file. *See* Cleraux, col. 5, ll. 56-60. If the requested file has filesystem characteristics that are the same as the operating system that requested the file, the file itself is sent. *See* Cleraux, col. 5, ll. 60-62. However, if the requested file has filesystem characteristics different from the operating system that requested the file, information from the database sufficient to emulate the requested file is sent to the operating system. *See* Cleraux, col. 5, l. 63 – col. 6, l. Cleraux further discloses that, in the latter instance, the emulation library may include information from the repository associated with the requested file but not the file itself. *See* Cleraux, col. 6, ll. 4-10.

In light of the above, Cleraux generally discloses sending information to emulate a file associated with a file characteristic of a first operating system as a file associated with a file characteristic of a second operating system. In this manner, Cleraux utterly fails to disclose or suggest converting anything, let alone converting a first filesystem type to a second filesystem type. Moreover, as Cleraux fails to disclose or suggest converting a first filesystem type to a second filesystem type, so too does Cleraux fail to disclose doing so concurrently with storing first and second anchor points.

Finally, claim 1 recites that the conversion "includ[es] activating the second directory and deleting the first directory." The Examiner admits on page 4 of the Office Action mailed January 15, 2010, that Cleraux fails to disclose activating the second directory and deleting the first directory. The Examiner relies on Jans to remedy this deficiency. Jans, however, discloses a method of hot swapping or migrating an application to a new version using migration plug-ins that facilitate the upgrading of the application while the application is still executing, with only

minor interruptions. Jans only briefly mentions directories as one of many types of entities that may exist in a data repository, so Applicant fails to appreciate how the Examiner can assert that Jans specifically discloses activating one directory while deleting another directory, particularly within the context of converting a filesystem from one type to another. Furthermore even assuming *arguendo* that Jans did disclose activating the second directory in the new filesystem and deleting the first directory in the old filesystem as contended by the Examiner, Jans still fails to remedy any of the other deficiencies of Cleraux identified above. Namely, Jans does not disclose storing in an object anchor points to first and second directories from first and second filesystems, or converting a filesystem from one type to another while maintaining full operational capacity concurrently with storing the anchor points.

In addition, the Examiner has provided no credible explanation as to why one of ordinary skill in the art would be motivated to modify Cleraux in the manner suggested by the Examiner. Cleraux, as noted above, is directed to emulating a filesystem for the purpose of allowing a device that does not have its own mass storage to access a filesystem on a host system, typically during development of an embedded application (col. 2, lines 3-10). In such applications, there would be no need to convert a filesystem from one type to another, and as such, one of ordinary skill in the art would not look to Cleraux for instruction on how to convert a filesystem from one type to another. Similarly, Jans is directed to hot swapping or migrating an application, and is completely silent with respect to the concept of converting a filesystem from one type to another. In fact, filesystems are mentioned as types of data repositories in paragraph [0031] of Jans, but are not enumerated as possible types of applications/services that could be migrated (also discussed in paragraph [0031]). Therefore, like Cleraux, Jans provides no motivation for modifying Cleraux to enable conversion of a filesystem from one type to another.

Therefore the combination of Cleraux in view of Jans fails to disclose or suggest all of the elements of Applicant's claim 1. Consequently, the Examiner has failed to establish a *prima facie* case of obviousness under the framework of the Graham factual inquiries because there is an unresolved difference between the subject matter of claim 1 and the disclosure in the references cited by the Examiner. Applicant respectfully requests that the Examiner's rejection of claim 1 be reversed.

Independent Claims 32 and 37

Independent claims 32 and 37 are program product versions of independent claim 1. Both claims 32 and 37 require and object having "a first anchor point that references the first directory, said first directory being of a first filesystem type" and "a second anchor point that references the second directory, said second directory being of a second filesystem type different than the first," similar to independent claim 1. Claims 32 and 37 also require program code "configured to convert the first filesystem type to the second filesystem type including activating the second directory and deleting the first directory while maintaining the filesystem in a full operational capacity," similar to independent claim 1. Therefore, for the same or similar reasons as set forth above with respect to claim 1, the combination of Cleraux in view of Jans fails to disclose or suggest all of the elements of Applicant's claims 32 and 37. Consequently, the Examiner has failed to establish a *prima facie* case of obviousness under the framework of the Graham factual inquiries because there is an unresolved difference between the subject matter of claims 32 and 37 and the disclosure in the references cited by the Examiner. Applicant respectfully requests that the Examiner's rejection of claims 32 and 37 be reversed.

Independent Claim 42

Independent claim 42 is an apparatus version of independent claims 1, 32, and 37. Claim 42 similarly requires program code "configured to store in the data structure a first anchor point for the object that references the first directory, said first directory implemented on a first filesystem type" and "store in the data structure a second anchor point for the object that references the second directory, said second directory implemented on a second filesystem type different than the first. Claim 42 also similarly requires that the program code is further configured to, "concurrently with storing the first and second anchor points, convert the first filesystem type to the second filesystem type including activating the second directory and deleting the first directory while maintaining the filesystem in a full operational capacity." Therefore, for the same or similar reasons as set forth above with respect to claims 1, 32, and 37, the combination of Cleraux in view of Jans fails to disclose or suggest all of the elements of Applicant's claim 42. Consequently, the Examiner has failed to establish a *prima facie* case of obviousness under the framework of the Graham factual inquiries because there is an unresolved difference between the subject matter of claim 42 and the disclosure in the references cited by

the Examiner. Applicant respectfully requests that the Examiner's rejection of claim 42 be reversed.

Dependent Claims 2-7

Dependent claims 2-7 are not argued separately.

Dependent Claim 28

Claim 28 adds to claim 1 that "the second filesystem type is a newer version of the first filesystem type." The Examiner contends that claim 28 is disclosed by Cleraux at col. 5, ll. 62-65. However, this passage of Cleraux merely states that if the request file is a regular file, then the emulation library will retrieve the file from a repository. If the file is a special UNIX file, such as a device file, the emulation library will retrieve information for that file from its internal database. There is no disclosure in this passage, nor anywhere else in Cleraux which discloses that the second filesystem is a newer version of the first filesystem. For this additional reason, the combination of Cleraux in view of Jans fails to disclose or suggest all of the elements of Applicant's claim 28. Consequently, the Examiner has failed to establish a *prima facie* case of obviousness under the framework of the Graham factual inquiries because there is an unresolved difference between the subject matter of claim 28 and the disclosure in the references cited by the Examiner. Applicant respectfully requests that the Examiner's rejection of claim 28 be reversed.

Dependent Claim 29

Claim 29 adds to claim 28 that "the second filesystem type is NTFS, and the first filesystem type is FAT32." The Examiner contends that claim 29 is disclosed by Cleraux at col. 1, ll. 29-50. However, this passage of Cleraux merely gives examples of different filesystem types for different operating systems. There is no disclosure in this passage, nor anywhere else in Cleraux which discloses that the second filesystem type is NTFS and the first filesystem type is FAT32. For this additional reason, the combination of Cleraux in view of Jans fails to disclose or suggest all of the elements of Applicant's claim 29. Consequently, the Examiner has failed to establish a *prima facie* case of obviousness under the framework of the Graham factual inquiries because there is an unresolved difference between the subject matter of claim 29 and the

disclosure in the references cited by the Examiner. Applicant respectfully requests that the Examiner's rejection of claim 29 be reversed.

Dependent Claim 30

Claim 30 is not argued separately.

Dependent Claim 31

Claim 31 adds to claim 30 that "the first filesystem type is associated with an HP-UX operating system, and the second filesystem type is associated with a Windows operating system." The Examiner contends that claim 31 is disclosed by Cleraux at col. 5, ll. 19-25. However, this passage of Cleraux merely states that the emulation library provides functions that act like common UNIX functions for a non-UNIX (Win32) filesystem type. There is no disclosure in this passage, nor anywhere else in Cleraux which discloses that the first filesystem type is associated with an HP-UX operating system, and the second filesystem is associated with a Window operating system. For this additional reason, the combination of Cleraux in view of Jans fails to disclose or suggest all of the elements of Applicant's claim 31. Consequently, the Examiner has failed to establish a *prima facie* case of obviousness under the framework of the Graham factual inquiries because there is an unresolved difference between the subject matter of claim 31 and the disclosure in the references cited by the Examiner. Applicant respectfully requests that the Examiner's rejection of claim 31 be reversed.

Dependent Claims 33-36

Dependent claims 33-36 are not argued separately.

Dependent Claim 38-41

Dependent claims 38-41 are not argued separately.

Dependent Claim 43-47

Dependent claims 43-47 are not argued separately.

CONCLUSION

Applicant respectfully requests that the Board reverse the Examiner's rejections of claims 1-7 and 28-47, and that the Application be passed to issue. If there are any questions regarding the foregoing, please contact the undersigned at 513/241-2324. If any other charges or credits are necessary to complete this communication, please apply them to Deposit Account 23-3000.

Respectfully submitted,

June 15, 2010
Date

/Charles R. Figer, Jr. /

Charles R. Figer, Jr.
Reg. No. 62,518
WOOD, HERRON & EVANS, L.L.P.
2700 Carew Tower
441 Vine Street
Cincinnati, Ohio 45202
Telephone: (513) 241-2324
Facsimile: (513) 241-6234

VIII. CLAIMS APPENDIX: CLAIMS ON APPEAL (S/N 10/777,870)

Listing of Claims:

1. (Previously Presented) A method for maintaining a data structure corresponding to an object having a first link from a first directory and a second link from a second directory in a filesystem, the object to which the data structure corresponds being selected from the group consisting of a file and a directory in the filesystem, the first and second directories being parent directories to the object to which the data structure corresponds, the method comprising the steps of:

storing in the data structure a first anchor point for the object that references the first directory, said first directory implemented on a first filesystem type;

storing in the data structure a second anchor point for the object that references the second directory, said second directory implemented on a second filesystem type different than the first; and

concurrently with storing the first and second anchor points, converting the first filesystem type to the second filesystem type including activating the second directory and deleting the first directory while maintaining the filesystem in a full operational capacity.

2. (Original) The method of claim 1, wherein the object is a file.

3. (Original) The method of claim 1, wherein the object is a directory.

4. (Previously Presented) The method of claim 3, wherein the directory is of the first filesystem type.

5. (Original) The method of claim 4, wherein the first link from the first directory to the object is a directory link; and the second link from the second directory to the object is a file link.

6. (Original) The method of claim 1, further comprising the steps of:

receiving a request for information about the first link; and

in response to the request, using the first anchor point when retrieving the information.

7. (Original) The method of claim 1, further comprising the steps of:

receiving a request for information about the object;

selecting the first anchor point instead of the second anchor point to respond to the request.

8-27. (Canceled).

28. (Previously Presented) The method of claim 1, wherein the second filesystem type is a newer version of the first filesystem type.

29. (Previously Presented) The method of claim 28, wherein the second filesystem type is NTFS, and the first filesystem type is FAT32.

30. (Previously Presented) The method of claim 1, wherein the first and second filesystem types are associated with different operating systems.

31. (Previously Presented) The method of claim 30, wherein the first filesystem type is associated with an HP-UX operating system, and the second filesystem type is associated with a Windows operating system.

32. (Previously Presented) A program product, comprising

a computer readable medium; and

a program code configured for maintaining a data structure corresponding to an object having a first link from a first directory and a second link from a second directory in a filesystem, the object to which the data structure corresponds being selected from the group consisting of a file and a directory in the filesystem, the first and second directories being parent directories to the object to which the data structure corresponds, the program code resident on the computer readable medium and further configured to store in the data structure a first anchor point for the object that references the first directory, said first directory implemented on a first filesystem type, store in the data structure a second anchor point for the object that references the second directory, said second directory implemented on a second filesystem type different than the first, and concurrently with storing the first and second anchor points, convert the first filesystem type to the second filesystem type including activating the second directory and deleting the first directory while maintaining the filesystem in a full operational capacity.

33. (Previously Presented) The program product of claim 32, wherein the object is a file.

34. (Previously Presented) The program product of claim 32, wherein the object is a directory.

35. (Previously Presented) The program product of claim 34, wherein the directory is of the first filesystem type.

36. (Previously Presented) The program product of claim 35, wherein the first link from the first directory to the object is a directory link, and the second link from the second directory to the object is a file link.

37. (Previously Presented) A program product, comprising

a computer readable medium;

a data structure configured to be maintained by an operating system and corresponding to an object having a first link from a first directory and a second link from a second directory in a filesystem, the object to which the data structure corresponds being selected from the group consisting of a file and a directory in the filesystem, the first and second directories being parent directories to the object to which the data structure corresponds, the data structure comprising:

a plurality of attributes related to the object;

a first anchor point that references the first directory, said first directory being of a first filesystem type; and

a second anchor point that references the second directory, said second directory being of a second filesystem type different than the first;

program code resident on the computer readable medium and configured upon execution to access the data structure, and concurrently with accessing the data structure, further configured to convert the first filesystem type to the second filesystem type including activating the second directory and deleting the first directory while maintaining the filesystem in a full operational capacity.

38. (Previously Presented) The program product of claim 37, wherein the object is a file.

39. (Previously Presented) The program product of claim 37, wherein the object is a directory.

40. (Previously Presented) The program product of claim 39, wherein the directory is of the first filesystem type.

41. (Previously Presented) The program product of claim 40, wherein the first link from the first directory to the object is a directory link, and the second link from the second directory to the object is a file link.

42. (Previously Presented) An apparatus comprising:

a processor; and

a program code configured to be executed by the processor to maintain a data structure corresponding to an object having a first link from a first directory and a second link from a second directory in a filesystem, the object to which the data structure corresponds being selected from the group consisting of a file and a directory in the filesystem, the first and second directories being parent directories to the object to which the data structure corresponds, the program code further configured to store in the data structure a first anchor point for the object that references the first directory, said first directory implemented on a first filesystem type, store in the data structure a second anchor point for the object that references the second directory, said second directory implemented on a second filesystem type different than the first, and concurrently with storing the first and second anchor points, convert the first filesystem type to the second filesystem type including activating the second directory and deleting the first directory while maintaining the filesystem in a full operational capacity.

43. (Previously Presented) The apparatus of claim 42, wherein the object is a file.

44. (Previously Presented) The apparatus of claim 42, wherein the object is a directory.

45. (Previously Presented) The apparatus of claim 44, wherein the directory is of the first filesystem type.

46. (Previously Presented) The apparatus of claim 45, wherein the first link from the first directory to the object is a directory link, and the second link from the second directory to the object is a file link.

47. (Previously Presented) The apparatus of claim 42, wherein the program code is further configured to:

select the first anchor point instead of the second anchor point to respond to a request for information about the object.

IX. EVIDENCE APPENDIX

[APPLICATION NO. 10/777,870]

None.

X. RELATED PROCEEDINGS APPENDIX

[APPLICATION NO. 10/777,870]

None.